**🎙️Slide 5–6: Validation & Test Results**

"Let’s now look at how each model performed — both during validation and on the final test set.

We started with **K-Means**, an unsupervised clustering method. On the validation set, it reached **52.47% accuracy**. But that’s barely better than guessing. On the test set, it dropped to **51.27%**. Why? Because K-Means doesn’t use sentiment labels — it just groups reviews based on similarity. That’s like trying to sort movies into genres without watching them — it might group ‘action’ and ‘horror’ together just because both mention ‘explosions’.

Next, we tested the **Perceptron**, a simple linear classifier. It performed quite well: **82% accuracy** on both validation and test sets. This tells us that the data is, to some extent, linearly separable — in other words, there’s a straight boundary between positive and negative reviews that this model can learn.

Finally, the **Multi-Layer Perceptron (MLP)** gave us the best results — **87% on validation**, and **85% on the test set**. That slight drop is expected and shows that our model didn't overfit. The MLP handles complex, non-linear relationships — and that’s important, because natural language is rarely straightforward.

In summary: **K-Means struggles without labels**, **Perceptron is reliable for simple cases**, and **MLP leads the way when complexity matters**."

**🎙️Slide 7–8: Confusion Matrix & Loss Curve**

"To dig deeper into the MLP’s performance, we analyzed its **confusion matrix**.

Out of all negative reviews, the model correctly labeled **637 as negative** — those are true negatives. It mistakenly labeled **94 as positive** — false positives.

For positive reviews, it got **634 right** — true positives — and mislabeled **135 as negative** — false negatives.

So, the model is slightly more cautious with positive reviews. It tends to misclassify them as negative, possibly because of sarcastic or ambiguous language. Still, with over **1,270 correct predictions out of 1,500**, its performance is solid.

We also looked at the **training loss curve** — this tells us how the error decreased during learning.

The plot shows that the loss dropped quickly and stabilized after just a few iterations. In fact, it converged after around **5 training iterations**, not 20 as initially stated. This is likely due to a mislabeling in the presentation — the **x-axis shows iterations**, not epochs.

Still, the takeaway is the same: the model quickly learned an efficient internal representation and didn’t overfit — thanks to **early stopping**, which stopped training once performance on the validation set stopped improving."

**🎙️Slide 9: Discussion**

"Let’s reflect on what we’ve learned.

First, **MLP clearly outperformed the other models**, confirming that deep neural networks are well-suited to sentiment analysis tasks. They can understand complex, non-linear patterns that simpler models miss.

The **Perceptron** also did quite well. It’s a good baseline and shows that many sentiment signals can be captured with linear decision boundaries — for example, words like *“love”* or *“terrible”* have strong, predictable effects.

On the other hand, **K-Means struggled**, as expected. Without label guidance, it couldn’t align clusters with actual sentiment. This is consistent with prior research and confirms the limitations of unsupervised learning in this context.

Our project reinforces three key ideas:

1. Supervised learning is essential for tasks like sentiment analysis.
2. Simple models can do a decent job, but struggle with nuance.
3. Neural networks shine when complexity and subtlety are involved."

Bonus:  
  
**📚Slide 10: References — Quick Overview**

Here’s a quick summary of each reference used:

1. **Maas et al. (2011)** — Source of the IMDb dataset.
2. **Bing Liu (2015)** — A foundational book on sentiment analysis theory and applications.
3. **Wang et al. (2016)** — Comparative analysis of different sentiment analysis algorithms; confirmed MLP’s strengths.
4. **Zhang et al. (2018)** — Survey of text classification methods; helped justify our approach.
5. **Joulin et al. (2016)** — Practical tricks for text classification; inspired preprocessing decisions.
6. **Bird et al. (2009)** — NLTK reference; source for stopwords and NLP preprocessing.
7. **Ramos (2003)** — Introduced TF-IDF usage in text relevance, which we adopted.
8. **Aggarwal (2012)** — Analysis of unsupervised text mining; aligned with our findings on K-Means.
9. **Joachims (1998)** — Discusses linear SVMs; conceptually supports the Perceptron.
10. **Kim (2014)** — CNNs for sentence classification; underpins the strength of neural models.
11. **Vaswani et al. (2017)** — Transformers; suggested as future direction.
12. **Pennington et al. (2014)** — GloVe embeddings; another future enhancement option.
13. **Blitzer et al. (2007)** — Explores **domain adaptation**, useful if we want to apply this model to other types of text beyond movie reviews.